(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number : **0 674 260 A2**

(12) EUROPEAN PATENT APPLICATION

(21) Application number : **95301835.5**

(51) Int. Cl.[6] : **G06F 9/46**

(22) Date of filing : **20.03.95**

(30) Priority : **21.03.94 US 215438**

(43) Date of publication of application :
27.09.95 Bulletin 95/39

(84) Designated Contracting States :
DE FR GB

(71) Applicant : **INTERNATIONAL BUSINESS MACHINES CORPORATION**
Old Orchard Road
Armonk, N.Y. 10504 (US)

(72) Inventor : **Kuo, Steve Tsanchun**
205 Manley Court
San Jose, California 95139 (US)
Inventor : **Morrison, Thomas Clarke**
899 Salt Lake Drive
San Jose, California 951333 (US)
Inventor : **Nguyen, Hoang Minh**
522 War Admiral Avenue
San Jose, California 95111 (US)
Inventor : **Radke, Harry Otto**
16 Cheltenham Way
San Jose, California 95139 (US)

(74) Representative : **Williams, Julian David**
IBM United Kingdom Limited,
Intellectual Property Department,
Hursley Park
Winchester, Hampshire SO21 2JN (GB)

(54) Transaction processing system.

(57) In a transaction processing system that includes a plurality of client processes (40) coupled to a server process (42), the server process supports execution of transactions generated by the client processes. The server process includes, for each client process being served, one or more transaction message control mechanisms. Each transaction message control mechanism includes a named processing object that includes a name identifying the object. The named processing object also includes an input process that receives all transaction request messages naming the object and identifying the originating client process. The input process dispatches a transaction process for each transaction request message received from the respective client process. Each transaction process oversees transaction execution and receives transaction output. A transaction process provides a transaction output message for the originating client process. In a non-sync'd mode of operation, transaction processes may synchronously send transaction output messages to client processes. In a synchronized mode of operation, one transaction process at a time sends output messages under control of an output process in the named processing object. The transaction message control mechanisms provide a bi-directional transaction message flow between server and client processes.

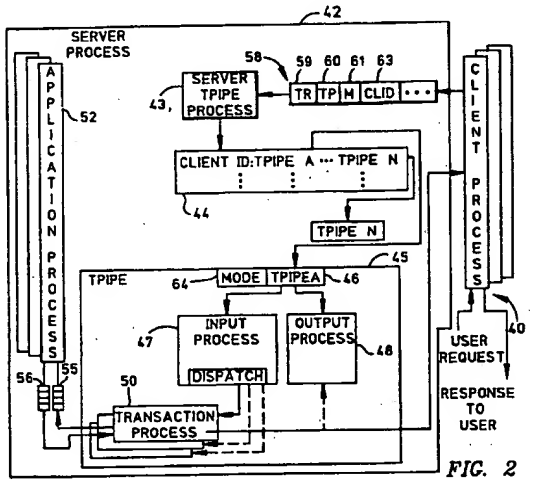

FIG. 2

EP 0 674 260 A2

The invention relates to transaction processing, and more particularly to a transaction processing system of the client/server type in which a client process creates a mechanism in a server process that controls bi-directional transaction message traffic between the client and server processes.

A transaction has been defined as a logical unit of work by C.J. Date in his work, INTRODUCTION TO DATABASE SYSTEMS, Volume I (Addison-Wesley, September, 1985). Date particularly taught that a transaction in the database context consists of a sequence of operations by which a database is transformed from one consistent state to another consistent state. A transaction processing system guarantees that a transaction will either complete the transformation of a database into a new consistent state, or return the database to the consistent state at which the transaction began.

A particularly useful architecture for implementing transaction processing in distributed systems is the client/server model that is described in H.M. Deitel's OPERATING SYSTEMS (Addison-Wesley, 1990). In this model, clients denote service consumers in the form of client processes, while servers are processes that provide the services. In a typical database system that provides transaction-based access to a plurality of users, users enter database requests through client processes. The client processes engage the server process in the form of a database management system to service the user request according to a predetermined transaction protocol.

From the standpoint of client and server processes, transactions may be considered as "objects". As objects, transactions encompass the procedures and data necessary to satisfy a user request. As objects, transactions can be directly manipulated by clients and servers. The preferred mode of handling transaction objects is by messages that specify a particular type of manipulation. In this regard, a user request for reading records from a database may manipulate a transaction by specifying that the transaction must read the database and by providing parameters that establish where the reading is to take place, what records are to be read, etc. The transaction reports the outcome of the manipulation by returning a response message including the results of the operation.

Typically, objects are described by data structures that are referred to as "names". Current techniques of object-oriented programming recognize a "named processing object" as an object having a name and including one or more procedures.

A transaction message implies a source and a destination. A current database management system, such as the IMS product available from INTERNATIONAL BUSINESS MACHINES CORPORATION, distinguishes these end points by the use of logical terminals (LTERMs). An LTERM provides a queue where transaction output from the IMS product resides. Eventually, the IMS product makes the connection between the queue and a physical node that will receive the queued output. In the client/server model, the IMS product is considered the server process. Such database systems do not afford a client process with a mechanism to control transaction message flow by specifying the source and destination of a transaction message.

Viewed from another aspect, the LTERM capability of the IMS product provides a uni-directional (server-to-client) pipeline that can only be manipulated by the server. In this respect, an LTERM suggests the pipe structure used in UNIX® systems (UNIX is a registered trade mark of Novell Inc.). Pipes may be objectified by names. They provide uni-directional, FIFO data transfer between processes and include the capability of interprocess synchronization. Two pipes oriented in opposite directions between client and server processes can provide bidirectional message flow, but entail two processing objects, only one of which can be manipulated by the client process.

Manifestly, the more time a server must spend in transaction message processing, the less efficient it will be in processing transactions. Accordingly, there is a need in transaction-based systems of the client/server type to provide a mechanism that will allow any client process to objectify a transaction and to manipulate the transaction object by messages in a bi-directional message pipeline. Viewed from one aspect the present invention provides a transaction processing system comprising:

a plurality of client processors for generating transaction request messages, each transaction request message including a request to execute a transaction and an object name;

a server process for executing transaction requests contained in transaction request messages from the client processes;

an interprocess message exchange facility coupled to the client processes and the server process for providing client process transaction request messages to the server process and providing transaction output messages to the client processes;

an input process in the named processing object for receiving all transaction request messages that include the object name and the identification at the respective client process; and

dispatch means in the input process for dispatching a transaction process in response to a transaction request message, each transaction process dispatched by the dispatch means being for receiving a transaction request message and for initiating a transaction for execution.